



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/700,181	11/03/2003	Ajay Arvind Apte	AUS919980908US2	3577

35525 7590 03/24/2005

IBM CORP (YA)  
C/O YEE & ASSOCIATES PC  
P.O. BOX 802333  
DALLAS, TX 75380

EXAMINER

MOFIZ, APU M

ART UNIT PAPER NUMBER

2165

DATE MAILED: 03/24/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

## Office Action Summary

Application No.

10/700,181

Applicant(s)

APTE, AJAY ARVIND

Examiner

Apu M Mofiz

Art Unit

2165

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

### Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

### Status

- 1) ☒ Responsive to communication(s) filed on 04 January 2005.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

### Disposition of Claims

- 4) ☒ Claim(s) 1-24 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-24 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

### Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 03 November 2003 is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

### Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
  - ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
  - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

### Attachment(s)

- |  |   |
|--|---|
| 1) <input checked="" type="checkbox"/> Notice of References Cited (PTO-892)  | 4) <input type="checkbox"/> Interview Summary (PTO-413)<br>Paper No(s)/Mail Date. _____ |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948)                                   | 5) <input type="checkbox"/> Notice of Informal Patent Application (PTO-152)             |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)<br>Paper No(s)/Mail Date _____ | 6) <input type="checkbox"/> Other: _____  |

## DETAILED ACTION

### ***Response to Applicant's Remarks***

1. In response to applicant's remarks, all previously presented rejections of claims are hereby withdrawn as to being moot.

### ***Claim Rejections - 35 USC § 102***

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

3. Claims 1-4, 7-9, 11-15, 18-20 and 22 are rejected under 35 U.S.C. 102(a) as being anticipated by Trevor et al. (The Use of Adapters to Support Cooperative Sharing, 1994 ACM, pages 219-230 and Trevor hereinafter).

As to claims 1,4,12,15,23 and 24, Trevor teaches a process for invoking a method of a server object (i.e., *"Interfaces describe the means by which a client interacts with the services provided by the object."*) The preceding text excerpts clearly indicate that the client interact with a server object through an interface." (page 222, col 1) in a distributed application in a distributed data processing system (page 219, col 1), the process comprising the computer-implemented steps of executing a client object (i.e., *"Dave et al. [7] extend the initial definition of proxy objects [22], as local representation of remote objects"*) The preceding text excerpts clearly indicate that the client object or the proxy object is the local representation of the server/target object. It is thru a proxy object that a client application interacts with the server/target object.) (page 221, col 2) in a client that implements a first programming environment (i.e., *"In supporting co-operation among a wide set of different users and*

*platforms, possibly with different interfaces to applications and even diverse language bindings, a uniform and clean event mechanism is needed.*" The preceding text excerpts clearly indicate that the client binds to a server object, which is of different language. Therefore a client object of language 1 interacts with server object of language 2.) (page 224, col 1), said client object (page 221, col 2) being written for said first programming environment (page 224, col 1); executing a server object (page 222, col 1) in a server that implements a second programming environment (page 224, col 1) that is different from said first programming environment (page 224, col 1), said server object being written for said first programming environment (page 224, col 1); executing (i.e., *"A client may use any adapter or object within the service, but to do so they must first request that the service bind them to the required object or adapter."* ... *"Successful completion of the binding process on an adapter object forms an invocation path between the client and the underlying object(s)."* The preceding text excerpts clearly indicate that the client requests the services from the underlying server objects. The services are requested only thru invocation of object methods.) (page 226, col 1) said client object (page 221, col 2) which begins an attempt to invoke a method (page 226, col 1) in the server object; in response to the client object beginning the attempt to invoke the method in the server object (page 222, col 1): obtaining an object reference (i.e., *"In order to create an object, a client passes an object template to the factory. The template is tested against the factories template list. If a successful match is made, the object, stored along the matched template, is created from its definition, and the resulting reference to the new object is returned to the client."* The preceding text excerpts clearly indicate that the client obtains an object reference/ client proxy by sending its template e.g. name to the factory.) (page 225, col 2) on the client (page 225, col 2) for said second programming environment (page 224, col 1); wrapping the object reference in an adapter (i.e., *"Interfaces describe the means by which a client interacts with the services provided by the object. Interface adapters provide facilities that abstract over these services to provide different services to users based upon context."* ... *"Adapters provide a level of indirection between object interfaces and object users. This level of*

*indirection allows interface adapters to provide additional facilities to manage the invocation of methods depending upon the context within which they are defined.*” ... *“At no point is an underlying object interface directly visible to a client, and the only way of invoking the operations is through an object adapter.”* ... *“Adapter A, Interface I adapts Object B, with Interface J This indicates that the adapter object A is an adapting object over object B, which may be a simple object or an adapter itself.”* The preceding text excerpts clearly indicate that the adapter adapts the object reference of the server or the adapted. The definition of an adapter can be found in the design pattern book of Gamma. The adapter by its definition adapts/ wraps the target object reference and thus provides the client of the adapter the services provided by the adapted object e.g., adapter A provides the services of adapted object B) (page 222, col 1, col 2; page 225, col 2) which isolates the object reference (i.e., the object B) (page 222, col 1, col 2; page 225, col 2) from said client object and the adapter performing data conversion (i.e., *“The basic structure of object adapters is as a set of mappings which map method names from those presented to users to those used within the object interface.”* The preceding text excerpts clearly indicate that the adapter maps/converts between the adapter interface and the actual server object interface. Therefore, a client’s invocation of an adapter method actually invokes a mapped server object method.) (page 222, col 2) between the object reference and the client object (page 221, col 2) for different types of data (page 224, col 1); calling by the client object (page 221, col 2) the method in the server object (page 222, col 1) wherein said step of calling invokes a method of the adapter (page 222, col 1, col 2; page 225, col 2), the method invoked in the adapter corresponding (i.e., *“The basic structure of object adapters is as a set of mappings which map method names from those presented to users to those used within the object interface.”* The preceding text excerpts clearly indicate that the adapter maps/converts between the adapter interface and the actual server object interface. Therefore, a client’s invocation of an adapter method actually invokes a mapped server object method.) (page 222, col 2) to the method called in the server object (page 222, col 1); calling, by the method invoked in the adapter (page 222, col 1, col 2; page 225, col 2), a method in the object reference that corresponds (page 222, col 2) to the method invoked in the adapter (page 222, col 1, col 2; page 225, col 2); and calling, by the object

reference, the method in the server object (page 222, col 1), wherein the adapter causes the object reference to invoke the method in the server object transparently to the client object such that the client object is unaware of the operation of the object reference (i.e., transparency of the target object reference method invocation is by definition of the adapter. This can be found in the design patterns book of Gamma, one of the inventors of the design patterns.) (page 222, col 1, col 2; page 225, col 2).

As to claims 2 and 13, Trevor teaches that the adapter (i.e., *"Interfaces describe the means by which a client interacts with the services provided by the object. Interface adapters provide facilities that abstract over these services to provide different services to users based upon context."* ... *"Adapters provide a level of indirection between object interfaces and object users. This level of indirection allows interface adapters to provide additional facilities to manage the invocation of methods depending upon the context within which they are defined."* ... *"At no point is an underlying object interface directly visible to a client, and the only way of invoking the operations is through an object adapter."* ... *"Adapter A, Interface I adapts Object B, with Interface J This indicates that the adapter object A is an adapting object over object B, which may be a simple object or an adapter itself."* The preceding text excerpts clearly indicate that the adapter adapts the object reference or the proxy of the server or the adapted object. The definition of an adapter can be found in the design pattern book of Gamma. The adapter by its definition adapts/ wraps the target object reference and thus provides the client of the adapter the services provided by the adapted object e.g., adapter A provides the services of adapted object B. By OMG definition of CORBA, the client object reference invokes method on the server skeleton. In this case, the client object reference is wrapped (or is an adapted of the adapting object) by an adapter. When a client application invokes a method on the adapter, the adapter in turn invokes the method of the object reference and the object reference has to invoke a method of the server skeleton by OMG CORBA definition.) (page 222, col 1, col 2; page 225, col 2) uses the object reference to invoke a method of a skeleton on the server (page 222, col 1, col 2; page 225, col 2).

As to claims 3 and 14, Trevor teaches that the skeleton invokes a method of a server object (i.e. by OMG CORBA standard definition) (page 222, col 1, col 2; page 225, col 2).

As to claims 7 and 18, Trevor teaches that the object reference is obtained from a naming service (i.e., *"In order to create an object, a client passes an object template to the factory. The template is tested against the factories template list. If a successful match is made, the object, stored along the matched template, is created from its definition, and the resulting reference to the new object is returned to the client."* The previous text excerpts clearly indicate that the client obtains an object reference/ client proxy by sending its template e.g. name to the factory. The factory is finding or creating the object reference from the name of the object.) (page 225, col 2).

As to claims 8 and 19, Trevor teaches that the proxy is a Common Request Broker Architecture (CORBA) proxy (page 221, col 1, col 2).

As to claims 9 and 20, Trevor teaches that the adapter calls a method of the CORBA proxy (i.e., *"Interfaces describe the means by which a client interacts with the services provided by the object. Interface adapters provide facilities that abstract over these services to provide different services to users based upon context."* ... *"Adapters provide a level of indirection between object interfaces and object users. This level of indirection allows interface adapters to provide additional facilities to manage the invocation of methods depending upon the context within which they are defined."* ... *"At no point is an underlying object interface directly visible to a client, and the only way of invoking the operations is through an object adapter."* ... *"Adapter A, Interface I adapts Object B, with Interface J This indicates that the adapter object A is an adapting object over object B, which may be a simple object or an adapter itself."* The preceding text excerpts clearly indicate that the

adapter adapts the object reference or the proxy of the server or the adapted object. The definition of an adapter can be found in the design pattern book of Gamma. The adapter by its definition adapts/ wraps the target object reference and thus provides the client of the adapter the services provided by the adapted object e.g., adapter A provides the services of adapted object B. By OMG definition of CORBA, the client object reference/ proxy invokes method on the server skeleton. In this case, the client object reference is wrapped (or is an adapted of the adapting object) by an adapter. When a client application invokes a method on the adapter, the adapter in turn invokes the method of the object reference/ proxy and the object reference/proxy has to invoke a method of the server skeleton by OMG CORBA definition.) (page 222, col 1, col 2; page 225, col 2).

As to claims 11 and 22, Trevor teaches that the CORBA proxy passes the method request to an object request broker (CORBA means Common Object Request Broker Architecture i.e., a client request passes thru the common ORB.) (page 221, col 1, col 2).

### ***Claim Rejections - 35 USC § 103***

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

5. Claims 5-6, 10 and 16-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Trevor et al. (The Use of Adapters to Support Cooperative Sharing, 1994 ACM, pages 219-230 and Trevor hereinafter) in view of Anne Thomas (Enterprise JavaBeans Technology, Patricia Seybold Group, Copy Right 1998, pages 1-24 and Thomas hereinafter).



The teachings of Trevor was discussed in the rejections above.

As to claims 5 and 16, Trevor teaches the adapter that implements an interface supported by the server object (i.e., *"Interfaces describe the means by which a client interacts with the services provided by the object. Interface adapters provide facilities that abstract over these services to provide different services to users based upon context."* ... *"Adapters provide a level of indirection between object interfaces and object users. This level of indirection allows interface adapters to provide additional facilities to manage the invocation of methods depending upon the context within which they are defined."* ... *"At no point is an underlying object interface directly visible to a client, and the only way of invoking the operations is through an object adapter."* ... *"Adapter A, Interface I adapts Object B, with Interface J This indicates that the adapter object A is an adapting object over object B, which may be a simple object or an adapter itself."* The preceding text excerpts clearly indicate that the adapter adapts the object reference or the proxy of the server or the adapted object. The definition of an adapter can be found in the design pattern book of Gamma. The adapter by its definition adapts/wraps the target object reference and thus provides the client of the adapter the services provided by the adapted object e.g., adapter A provides the services of adapted object B. By OMG definition of CORBA, the client object reference/proxy invokes method on the server skeleton. In this case, the client object reference is wrapped (or is an adapted of the adapting object) by an adapter. When a client application invokes a method on the adapter, the adapter in turn invokes the method of the object reference/proxy and the object reference/proxy has to invoke a method of the server skeleton by OMG CORBA definition.) (page 222, col 1, col 2; page 225, col 2).

Trevor does not explicitly teach Java as the language for the interface.

Thomas teaches Java as the language for the interface (i.e., a Java client application invokes a method over RMI. RMI by definition is all Java interface.) (page 4).

It would have been obvious to a person of ordinary skill in the art at the time of Applicant's invention to modify the teachings of Trevor with the teachings of Thomas to include Java as the language for the interface with the motivation to use the Java technology APIs because a developer can implement an application system that makes

use of whatever enterprise services happen to exist on the platform in use (Thomas, page 4, paragraph 1).

As to claims 6 and 17, Trevor does not explicitly teach that the server object is an Enterprise JavaBean.

Thomas teaches that the server object is an Enterprise JavaBean (i.e., "*For example, Money Makers, a large brokerage house, is using Enterprise JavaBeans technology to implement an application system to manage stock fund accounts.*") (page 2; page 4).

It would have been obvious to a person of ordinary skill in the art at the time of Applicant's invention to modify the teachings of Trevor with the teachings of Thomas to include that the server object is an Enterprise JavaBean with the motivation to use EJB because EJB is highly scalable, highly available, highly reliable, highly secure, transactional, distributed application (Thomas, page 3, paragraph 5).

As to claims 10 and 21, Trevor teaches a CORBA proxy residing on a client computer (page 221, col 1, col 2).

Trevor does not explicitly teach Java as the language for the interface.

Thomas teaches Java as the language for the interface (i.e., a Java client application invokes a method over RMI. RMI by definition is all Java interface.) (page 4).

It would have been obvious to a person of ordinary skill in the art at the time of Applicant's invention to modify the teachings of Trevor with the teachings of Thomas to include Java as the language for the interface with the motivation to use the Java

Art Unit: 2165


technology APIs because a developer can implement an application system that makes use of whatever enterprise services happen to exist on the platform in use (Thomas, page 4, paragraph 1).

***Points of Contact***

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Apu M. Mofiz whose telephone number is (571) 272-4080. The examiner can normally be reached on Monday – Thursday 8:00 A.M. to 4:30 P.M.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Dov Popovici can be reached at (571) 272-4083. The fax numbers for the group is (703) 872-9306.

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-9600.



Apu M. Mofiz  
Patent Examiner  
Technology Center 2100

March 08,2005